DOI: 10.1111/cgf.70331

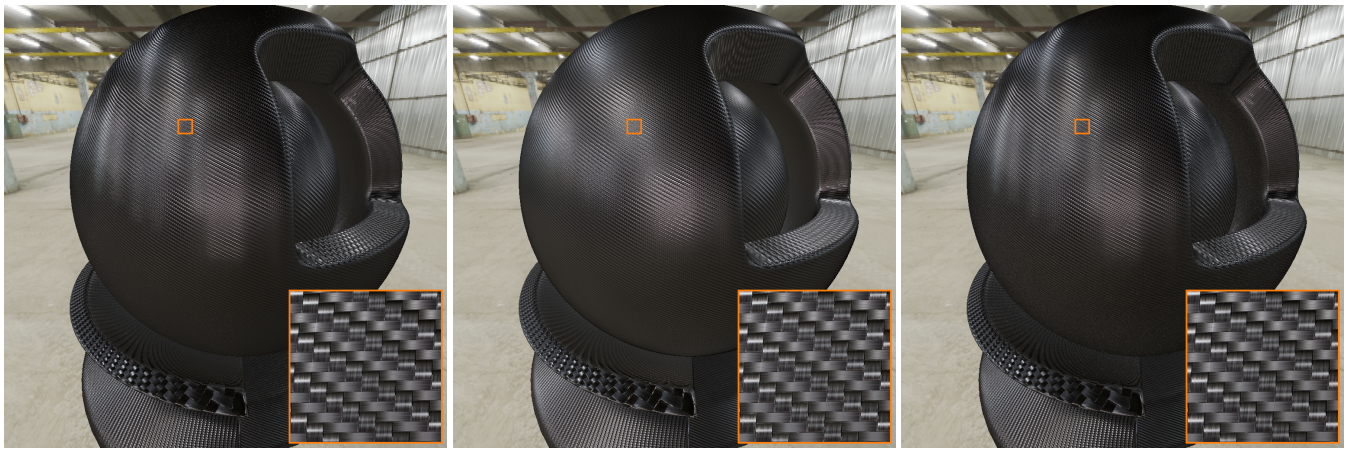
EUROGRAPHICS 2026 / B. Masia and J. Thies
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 2

A Real-Time Multi-Scale Neural Representation for Complex Surface Reflectance

Heikki Timonen[1], Pauli Kemppinen[1] and Jaakko Lehtinen[1,2]

[1]Aalto University, Finland [2]Nvidia Research, Finland



Ground Truth GGX Ours

Figure 1: *Rendering of an object that features an extremely high-frequency SVBRDF (see close-up). Spatial antialiasing turns the complex surface variations into an apparent BRDF of high angular complexity. Representing this complexity is impossible for classic analytic models and difficult for modern learned approaches. Our learning architecture, based on sampling and interpolation across the space-scale-angle domains, yields significantly better results at computational cost within an order of magnitude from analytic models.*

Abstract

Recent machine learning methods have significantly advanced the state of the art in the classic problem of representing surface appearance over angle, space, and scale. The models tend, however, to be relatively heavy compared to traditional fixed-function representations, making real-time application challenging. We present a neural shading architecture that allows the use of smaller and faster-to-evaluate neural networks than current state of the art, while faithfully representing complex spatial and angular variation. We target the angular complexity that arises both from prefiltering normal-mapped SVBRDFs, as well as complex, measured homogeneous BRDFs. A key architectural innovation is the introduction of a multiplicative interaction ("gating") between learnable parameters that significantly increases our model's expressive power. Our straightforward, unoptimized shader implementation renders over 1000 full HD frames per second on a consumer GPU using our default parameters.

CCS Concepts

• ***Computing methodologies*** → ***Reflectance modeling;*** *Neural networks;*

1. Introduction

Photorealistic replication of surface appearance is a key problem in computer graphics. The light–surface interaction is typically modeled with the (possibly Spatially Varying) Bidirectional Reflection Distribution Function or (SV)BRDF, which represents the distribution of directions of scattering given an inbound direction of light at a given surface location.

Based on assumptions about the microgeometry and reflective properties of a surface, many physically-based reflectance models with relatively simple closed-form expressions have been de-

rived [CT82, WMLT07]. Many real-life materials have, however, characteristics more complex than what can be captured by such models, yielding poor numerical fits [NDM05]. While, in principle, BRDFs could be stored as high-dimensional arrays, tabulation becomes infeasible in almost any practical application without compression due to enormous storage requirements [DJ18]. The problem is exacerbated in SVBRDFs, where spatial prefiltering may reveal complex angular behavior when viewed from afar, even if the finest resolution consists of a simple parametric SVBRDF model (cf. Figure 1).

Recently, neural material models [SRRW21, RJGW19, KMX*21, ZRW*24] have been shown to be able to represent complex reflectance functions, including spatial variation and the effects of prefiltering. However, most neural models are computationally significantly heavier than even the most sophisticated analytical models, hindering application in particularly real-time scenarios. Neural appearance models targeting real-time applications have only recently been introduced [ZRW*24].

The focus of this work is developing a neural reflectance model that is computationally lightweight enough for real-time applications, and yet faithfully able to model complex angular effects found in both measured homogeneous BRDFs and created by prefiltering SVBRDFs. We leave refraction and importance sampling out of scope, and focus discussion on RGB modeling instead of full spectral dependence. While many similarities exist, we focus on surface reflectance and leave materials with complex mesostructures of the kind treated by Bidirectional Texture Function (BTF) research out of scope. Finally, as computational efficiency is a primary goal, we focus on overfitting individual materials rather than on their synthesis, interpolation or editing [XMWY24, RPKLMG23, RGJW20].

1.1. Overview and Contributions

We assume that we are given as input an underlying, potentially spatially varying reflectance function $f_r(\omega_i, \omega_o | u, v)$, where ω_i, ω_o are the incident and outgoing directions and u, v are spatial coordinates in a (planar) 2D domain. For fixed u, v, the remaining degrees of freedom ω_i, ω_o comprise a standard BRDF. We do not assume any particular form to this (SV)BRDF, only that we can evaluate it pointwise over continuous coordinates.

In the case of spatially homogeneous materials (no dependence on u, v), we seek to accurately represent complicated behavior over ω_i, ω_o, as expressed in many measured BRDF models [MPBM03, DJ18].

In the case of spatially varying materials, we seek to efficiently approximate the result of rendering the underlying surface with perfect antialiasing. Assuming the incident and outgoing directions are locally constant (that we view and illuminate the surface from relatively far away), that the surface is not highly curved relative to the pixel footprint, and that the whole pixel footprint is covered by the same surface, we can rewrite the standard anti-aliasing convolution over screen pixels with filter w as a convolution with an anisotropic filter $w_\Sigma(u, v)$ in texture space:

$$f_r(\omega_i, \omega_o | u, v, \Sigma) = \int_{s,t} w_\Sigma(u-s, v-t)\, f_r(\omega_i, \omega_o | s, t)\, \mathrm{d}s\, \mathrm{d}t, \quad (1)$$

$$\text{with } w_\Sigma(u, v) = w(\Sigma\,(u\ v)^\perp)/\det\Sigma.$$

Here the 2×2 matrix Σ encodes the local size, orientation, and elongation of the filter, and is obtained by locally linearizing perspective projection [Hec89].

Equation 1 mixes the angular BRDFs of nearby surface points into *apparent BRDFs*. Interestingly, even if the angular behavior of the underlying reflectance function is simple for each individual surface point, the local averaging can, particularly when the underlying surface features sharp gloss and normal variations, lead to arbitrarily complex behaviors (see Figure A.12). In contrast, when the underlying surface is purely diffuse and has no normal variations, these apparent BRDFs reduce to simple averages of albedos; a vast literature on texture filtering treats this case.

Contributions. We present a model for $f_r(\omega_i, \omega_o | u, v, \Sigma)$ with the ability to accurately represent and efficiently render, in real time, a large class of complex surface materials with full support for anisotropic filtering. Based on neural networks, our main contribution is a neural architecture tailored to the specific challenges of representing complex angular effects both in measured materials and arising from spatial prefiltering. Specifically, we introduce a multiplicative interaction between learned angular and spatial features, and a learned warping that allows better use of finite texture space for angular features.

2. Related work

2.1. Analytic BRDF models and SVBRDFs

For most of the history of computer graphics, the angular dependence in the reflectance function $f_r(\omega_i, \omega_o | u, v, \Sigma)$ has been represented by parametric models with fast, closed-form expressions and increasing degrees of physical plausibility and intuitive control [Gou71, Pho75, Bli77, CT82, ON94, WMLT07, War92, AS00, LFTG97, APS00, BSH12, LKYU12, BS12]. Spatial variation in surface reflectance is traditionally represented by changing the parameters of an analytic BRDF model as a function of space through texture mapping:

$$f_r(\omega_i, \omega_o | u, v, \Sigma) = f_r(\omega_i, \omega_o | T(\mathcal{S}, u, v, \Sigma)). \quad (2)$$

Here $\mathcal{S}$ is a texture map (or mip-map pyramid) that stores parameters for the analytic angular model and T is the texture mapping operation that returns a vector interpolated from the texels of $\mathcal{S}$ to the position u, v, applying mip-mapping or anisotropic filtering if a mip-map pyramid is available. It is noteworthy that this approach treats the angular and spatial domains using very different mechanisms: the angles through an analytic formula and space through tabulation and interpolation.

BRDF Parameterization. Rusinkiewicz [Rus98] presents an elegant parameterization for the angular arguments of the BRDF. Instead of raw incident and outgoing directions, the parameterization consists of the half-vector ω_h between ω_i and ω_o and their difference ω_d. As noted by Rusinkiewicz, these coordinates greatly simplify the representation of many reflectance functions by aligning

Table 1: *Central differences between our model and closest related work when representing SVBRDFs. *The Tripleplane method supports synthesis in many scales but is not directly aware of prefiltering over scale.*

Model	Task	LoDs	Angular representation	Parametrization	Shading frames [ZRW*24]	Adaptive resolution	Spatial conditioning
Naive MLP	overfit	no	MLP	direct	n/a	n/a	n/a
NBRDF [SRRW21]	overfit	no	MLP	Rusinkiewicz	n/a	n/a	n/a
Zeltner [ZRW*24]	overfit	isotropic	MLP	direct	yes	no	concatenate
NeuMIP [KMX*21]	overfit	isotropic	MLP	direct	no	no	concatenate
Dou & Zheng [DZJ*24]	overfit	no	ω_h, ω_d sparsely tabulated + MLP	direct	no	no	concatenate
Biplane [FWH*23]	BTF compr.	no	ω_h densely tabulated + MLP	Rusinkiewicz	no	no	concatenate
Tripleplane [XMWY24]	BTF synth.	no*	ω_h, ω_d densely tabulated + MLP	Rusinkiewicz	no	no	concatenate
Ours	overfit	full aniso	ω_h, ω_d densely tabulated + MLP	Rusinkiewicz	yes (Sec. 3.1)	yes (Seq. 3.2)	gating (Eq. 6)

specular highlights to the center of the half-vector domain regardless of the orientations of the original ω_i and ω_o. Furthermore, this representation makes it easy to enforce the physical constraint of reciprocity (invariance to swapping ω_i and ω_o), which manifests as π-periodicity in the azimuthal angle of the spherical coordinate of the difference vector. We will make use of this parameterization.

2.2. Classical reflectance prefiltering

Shading of complex spatially varying surfaces is, in general, nonlinear with respect to the parameters of the shading model [BN11]. Hence, it is usually not possible to acquire the correct filtered image, with the reflectance function defined as in Equation 1, by first linearly filtering the spatial parameter maps that drive parametric BRDFs, as in Equation 2.

Originally reflectance prefiltering treated the normal variation due to bump maps in particular [Bli78]. Some early representations for the reflection shape were spherical harmonics [CMS87], sums of lobes [Fou92], and blends between these and displacement mapping based on the viewing condition [BM93].

More recently Han et al. [HSRG07] tackle the general filtering problem as convolution on the hemisphere and approximate the effective (filtered) BRDF with either a spherical harmonics expansion or a mixture of von Mises–Fisher distributions. Xu et al. [XWZB17] propose a real-time variant, also based on a von Mises–Fisher mixture. Wu et al. [WDR11] compute a discretization of visibility of small-scale geometry to approximate filtered reflectance in real time.

Many methods attempt to find principled means for generating a new set of parameter maps that approximate a filtered appearance using the same underlying parametric BRDF [Tok05, OB10, DHI*13]. Recently, deep-learning based methods have also emerged for optimizing spatial mip-map pyramids for SVBRDFs while still using analytic models for the angular variation [GFL*22, KAL23].

2.3. Neural material models

A growing literature uses neural techniques also for representing the angular variation in reflectance functions. A prototypical setup is to use a multilayer perceptron (MLP) that takes as input the directions ω_i, ω_o and a latent vector interpolated from a learned texture map $\mathcal{S}$:

$$f_r(\omega_i, \omega_o | u, v, \Sigma, \theta, \mathcal{S}) = \mathrm{MLP}_\theta(\omega_i, \omega_o, T(\mathcal{S}, u, v, \Sigma)). \quad (3)$$

Here θ are the network's parameters that are optimized jointly with the latent codes stored in the texels $\mathcal{S}$ by gradient-based methods to minimize the discrepancy of the MLP's output and the desired reflectance. The work of Kuznetsov et al. [KMX*21] and Zeltner et al. [ZRW*24] take this approach. Both techniques also account for prefiltering over scale by creating a mip-map pyramid of latent codes, but do not consider anisotropy. To represent parallax effects common in BTFs, Kuznetsov et al. modify T with a neural offset module that is conditional on ω_i. Conversely, Zeltner et al. modify ω_i and ω_o by a local shading frame conditional on the spatial latent. Notably, the overall structure of these techniques remains similar to the classic approach from Equation 2: textures for space and scale, and a single, "monolithic" model for the angles.

When applied to highly complex reflectance functions, the pure MLP used for representing angular effects struggles due to MLPs' well-known low-frequency bias [CFW*21]. This has led several authors to introduce tabulation (texture mapping) also for modeling the angular effects thereby trading a small increase in memory to more locality in the angular model's learning task. Working in Rusinkiewicz coordinates ω_h, ω_d, the "neural biplane model" of Fan et al. [FWH*23] uses latent texture maps $\mathcal{S}$ and $\mathcal{H}$ over space and the half-vector ω_h, and represents dependence over ω_d with an MLP:

$$f_r(\omega_i, \omega_o | u, v, \Sigma, \theta, \mathcal{S}, \mathcal{H}) = \mathrm{MLP}_\theta(\omega_d, T(\mathcal{H}, \omega_h), T(\mathcal{S}, u, v, \Sigma)) \quad (4)$$

This model was extended to the "tripleplane model" [XMWY24] and its quantized variant [XCL*25] that use latent textures also for the dependence on the difference vector ω_d:

$$f_r(\omega_i, \omega_o | u, v, \Sigma, \Theta) = \mathrm{MLP}_\theta\left(\overbrace{T(\mathcal{D}, \omega_d)}^{\omega_d \text{ features}}, \underbrace{T(\mathcal{H}, \omega_h)}_{\omega_h \text{ features}}, \overbrace{T(\mathcal{S}, u, v, \Sigma)}^{\text{space-scale features}}\right), \quad (5)$$

where we jointly define $\Theta \equiv [\theta, \mathcal{S}, \mathcal{H}, \mathcal{D}]$. Notably, the MLP now only takes as input the interpolated features for the half-vector ω_h, the difference vector ω_d, and the space-scale features — not the angles themselves. While aimed at synthesizing or compressing BTFs instead of representing surface reflectance, these models are architecturally closely related to ours. However, they do not account for prefiltering over scale.

Closely related to the biplane and tripleplane models, Dou et al. [DZJ*24] present a model where angular features are sparsely tabulated on the unit sphere (instead of densely on a planar grid). While their approach is conceptually elegant, they need to resort to

quantized codebook compression to maintain a feasible parameter count. They also apply piecewise constant interpolation of angular features during model evaluation for computational efficiency which potentially limits visual fidelity.

Table 1 compares our approach with the closest related work along several axes.

We note in passing that a similar development of using more spatially indexed storage in lieu of coordinate-based MLPs has taken place in neural view interpolation research: after the initial success of NeRF [LGZL*20], several authors realized that storing features in space often leads to better results and faster convergence [MESK22, CXG*22, HX23, XXP*23]. Recently, Wu et al. [WBX*24] successfully extended the spatial feature grid encoding to the angular domain in novel view synthesis. They model ideal specular reflections using learned features stored in a cube map. We also note some similarity to the work of Datta et al. [DMD*23]. They device a scheme of "differentiable indirection" where an array of interpolated indices ("pointers") is used to interpolate a secondary array of task-specific values, for example values of a BRDF for reflectance or a signed distance function (SDF) for geometry representation. They, however, only test their method with relatively simple analytic reflectance models.

2.4. Neural importance samplers

Importance sampling is a widely adopted technique for variance reduction in Monte Carlo integration. However, efficient and exact importance sampling of reflectance functions is not always possible even with analytical BRDF models and the sampling function is merely an approximation of the underlying reflectance function. This remains an open problem also with neural reflectance models; the importance sampler is an approximation and almost always an independently trained model, separate from the reflectance model. Prior work either predicts parameters of an analytical sampler [SRRW21, ZRW*24], uses a normalizing flow or a diffusion model [FBL*24, XWH*23] or directly warps samples without explicit density evaluation [WBX*25, BWZ*25].

3. Method

Our model is closely related to the "tripleplane" model [XMWY24](Equation 5) in that we use the Rusinkiewicz coordinates and learned latent texture maps over the half vector, difference vector, and space-scale coordinates. Deviating from their approach, we condition the model by allowing the spatial and angular features to interact multiplicatively before being input to the final MLP. This significantly increases the model's expressive power.

Concretely, we store matrices instead of vectors in the latent textures $\mathcal{H}$ and $\mathcal{D}$, and the two angular latent vectors are formed by multiplying the interpolated matrices by the spatial latent vector z_s; further, we drop the spatial latent from the decoder MLP's inputs, relying *only* on the gating mechanism for spatial conditioning:

$$\begin{gathered}\ln f_r(\omega_\mathrm{h},\omega_\mathrm{d}|\,u,v,\Sigma,\Theta)\approx \mathrm{MLP}_\theta(z_\mathrm{h},z_\mathrm{d}),\quad \text{with} \\ z_\mathrm{h}=\mathbf{H}z_\mathrm{s},\quad z_\mathrm{d}=\mathbf{D}z_\mathrm{s}, \\ \mathbf{H}=T(\mathcal{H},\omega_\mathrm{h}),\quad \mathbf{D}=T(\mathcal{D},\omega_\mathrm{d}),\quad \text{and } z_\mathrm{s}=T(\mathcal{S},u,v,\Sigma).\end{gathered} \tag{6}$$

Table 2: *Table of symbols.*

ω_i	Incident (lighting) direction
ω_o	Outgoing (viewing) direction
ω_h	Half-vector between ω_i and ω_o
ω_d	Difference vector between ω_i and ω_o [Rus98]
u,v	Spatial texture map coordinate
σ	Scale coordinate (MIP map level)
Σ	2×2 anisotropic filter footprint matrix
$\mathcal{S}$	mip map of latent vectors over u,v and σ, contains z_s
$\mathcal{H}$	$H_\mathrm{h}\times W_\mathrm{h}$ latent texture over half-vectors ω_h, contains $\mathbf{H}$
$\mathcal{D}$	$H_\mathrm{d}\times W_\mathrm{d}$ latent texture over diff. vectors ω_d, contains $\mathbf{D}$
z_s	Spatial latent vector $\in\mathbb{R}^{D_\mathrm{s}}$
z_h	Half-vector latent $\in\mathbb{R}^{D_\mathrm{h}}$
z_d	Difference vector latent $\in\mathbb{R}^{D_\mathrm{d}}$
$\mathbf{H}$	Half-vector latent matrix $\in\mathbb{R}^{D_\mathrm{h}\times D_\mathrm{s}}$, stored in $\mathcal{H}$
$\mathbf{D}$	Difference vector latent matrix $\in\mathbb{R}^{D_\mathrm{d}\times D_\mathrm{s}}$, stored in $\mathcal{D}$
D_s	Dimension of spatial latent vector z_s
D_h	Dimension of the half-vector latent
D_d	Dimension of the difference vector latent
D_in	Dimension of the decoder MLP's input

Figure 2 illustrates the architecture. Note, that like several prior works, the MLP outputs the logarithm of the reflectance function so that the final value is obtained by exponentiation.

The motivation for the multiplicative mechanism is to enable a learned "choice" or "gating" mechanism for the spatial latent vectors' interaction with the angular latent vectors, instead of having both be mixed by the MLP. We speculate that since the angular texture is shared between all spatial locations, multiplication allows the model to choose the most important features for the local task from a shared angular codebook. (We note in passing that multiplicative interactions between learned components have lead to significant advances in a variety of other contexts, including the "style mechanism" in StyleGAN [KLA19] and, of course, the ubiquitous attention mechanism [BCB15].)

We access the angular textures $\mathcal{H},\mathcal{D}$ using polar coordinates of ω_h and ω_d projected onto the unit disk.

$$r=\|\omega_\perp\|,\quad \varphi=\mathrm{atan2}\left(\omega_\perp^y,\omega_\perp^x\right),$$

where $\omega_\perp$ is a disk-projected half or difference vector. We ensure symmetry of the resulting BRDF (invariance to swapping $\omega_\mathrm{i}\leftrightarrow\omega_\mathrm{o}$) by wrapping the texture reads $T(\mathcal{D},\omega_\mathrm{d})$ of the difference vector to the interval $[0,\pi]$ in the φ direction. We use circular padding in the φ direction during angular texture reads to avoid discontinuities and repeat padding in the r direction, broadcasting the first element in r direction for all φ.

Anisotropic filtering. Because the filter shape Σ is visible to the MLP only indirectly in the contents of the spatial latent z_s, we note that it is trivial to introduce support for anisotropy without architectural changes. This is achieved simply by extending the training distribution to cover anisotropies and performing the texture fetch from the mip map pyramid $\mathcal{S}$ using anisotropic filtering during both model fitting and inference. Of course, the ground truth training targets need to be computed by employing the correspond-

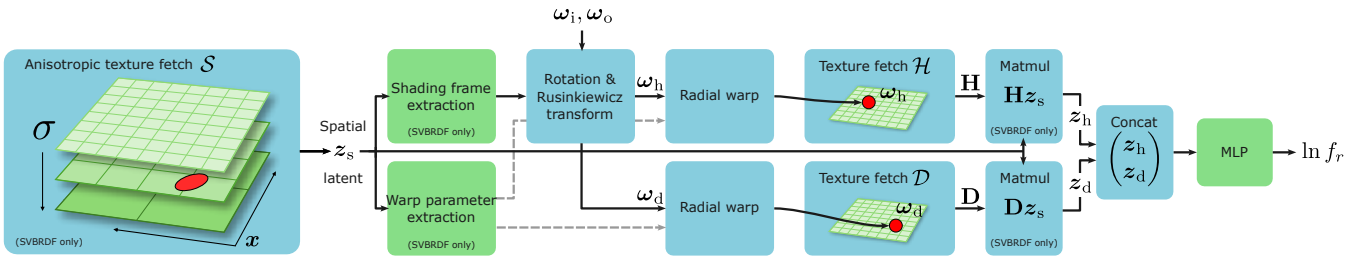


Figure 2: *Our architecture (see Section 3). The leftmost spatial texture fetch and the matrix multiplications with* $\mathbf{H}, \mathbf{D}$ *are skipped when modeling spatially homogeneous materials, as the angular textures* $\mathcal{S}, \mathcal{D}$ *directly contain the angular latents* z_h *and* z_d*.*

ing anisotropies. Our results show the expected kinds of improvements from this approach.

3.1. Shading Frames.

The Rusinkiewicz coordinates leave open the question of how the local coordinate frame — in which ω_i, ω_o and consequently ω_h, ω_d are expressed — aligns with the texture space. While the simplest solution would be to use a mapping whose tangent and binormal coincide with the texture *uv* axes and the normal always points straight up, we find, like previous work, that performance is increased by a more judicious choice of coordinates that allows simpler representations for effects such as normal mapping.

Specifically, we adopt the learned "shading frames" of Zeltner et al. [ZRW*24] as means of feature alignment, whereby a normal and tangent vector are extracted from the interpolated spatial latent z_s with a learned linear mapping. The input coordinates ω_i, ω_o are transformed to this coordinate system before computing the Rusinkiewicz coordinates ω_h, ω_d. Different from the original implementation, we project ω_i, ω_o to the unit disk immediately after applying the frame (and employ $N = 1$ frames) before lifting the directions back to the unit hemisphere again in preparation for computing the Rusinkiewicz coordinates. This projection avoid situations where a direction would end up under the horizon after the application of a shading frame. The shading frame component is disabled for materials with no spatial variation.

3.2. Radial Warps in Angular Textures

We observe empirically, and in line with prior work [DJ18], that the need for resolution in the angular textures $\mathcal{H}$ and $\mathcal{D}$ may be highly non-uniform. We therefore insert simple parametric, learnable warps to the continuous ω_h, ω_d coordinates used to index and interpolate the respective angular textures. This effectively gives the model means of concentrating resolution to areas of the angular space where changes in the target function are rapid. We choose a simple polynomial warp in the radial direction of the form

$$r' = \sum_k w_k r^{\alpha_k}, \tag{7}$$

where w_k are a learned partition of unity and $\alpha_k \in (0, \infty)$ are learned parameters. In spatially varying BRDFs, the parameters are decoded from the spatial latent z_s via a single learned linear layer. We use two components in the sum in all our experiments and bound each α_k from below by 1/2. We initialize the warp to be approximately the identity in the beginning of training. Both angular textures $\mathcal{H}$ and $\mathcal{D}$ have their own dedicated warps.

We recover the choice of Dupuy and Jacob [DJ18] by the choice $k = 1, w_1 = 1, \alpha_1 = 1/2$. The radial warps are particularly useful when the variation of the target BRDF is aligned with the center (e.g. specular highlight in the half vector space) or the outer boundary (Fresnel effect) of each coordinate system, which is often the case with the Rusinkiewicz parametrization.

4. Training details

4.1. Data sets and preparation

We train our models using measured reflectance data from the MERL database [MPBM03] that contains 100 spatially uniform isotropic BRDFs ranging from diffuse fabrics to specular metals. In addition to MERL materials we also train on materials from the RGL dataset [DJ18], which also contains some anisotropic materials, albeit still spatially uniform. We handpick a subset of 30 materials, focusing mostly on specular materials with interesting angular behavior.

We use the MatSynth dataset [VD24] as the source of spatially-varying materials. It consists of thousands of high-resolution, high-diversity SVBRDFs. We pick a subset of 25 materials that exhibit particularly interesting behavior under spatial filtering. We define the underlying SVBRDF in Equation 1 through bilinearly interpolating the highest-resolution parameter maps to the required u, v and shading the result using the BRDF model. (Note that we *do not* employ mip-maps here, as the function is supposed to retain all of its complexity before being filtered spatially as per Equation 1.) To keep training times manageable, we downsample the ground truth MatSynth parameter maps to 1024^2.

We fit our models to only the BRDF and divide out any possible cosine term (e.g. the RGL reference implementation evaluates BRDF $\times$ cosine). We symmetrize each target using $\hat{f}_r = (f_r(\omega_i, \omega_o) + f_r(\omega_o, \omega_i))/2$, which is a no-op for physically plausible BRDFs.

4.2. Training task

We train a single model per material by minimizing the difference of the model prediction and ground truth over all spatial locations

u, v, anisotropic prefilter scales and shapes Σ, and incident-outgoing direction pairs ω_i, ω_o:

$$\min_{\Theta} \mathbb{E}_{\omega_o,\omega_i,\Sigma,u,v,} \left[L(\omega_o, \omega_i, u, v, \Sigma, \Theta) \right], \tag{8}$$

We use a mixture of prediction-relative L_2 and pure L_2 as the loss function L (Section 4.4). We draw the directions ω_o, ω_i with cosine weighted density with respect to the geometric normal and the spatial coordinates u, v uniformly on the unit square.

We draw the scalar scale of the Σ by choosing finer mip levels more often than coarser levels. We construct the matrix Σ by

$$\Sigma = e_1 \begin{pmatrix} (1+z_2)/\cos(\theta_i) & 0 \\ 0 & 1 \end{pmatrix} R_{u_3}, \tag{9}$$

where $\cos(\theta_i)$ is the cosine of the angle between the geometrical normal and the viewing direction. Furthermore, we choose $\log_2(e_1) = \ell_{\min} + (\ell_{\max} - \ell_{\min})e_2$, with $\ell_{\min} = -13$, $\ell_{\max} = 1$. We weigh fine levels of the mip pyramid more by choosing $e_2 = (\exp(5e_3) - 1)/(\exp(5) - 1)$, and $e_3 \sim \mathcal{U}(0,1)$. The remaining values are drawn as $z_2 \sim \mathcal{N}(0, 0.2^2)$, and $u_3 \sim \mathcal{U}(0, 2\pi)$, R_{u_3} is a 2D rotation of angle u_3 along the geometric normal. This construction ensures that when viewed from grazing angles, more anisotropic configurations are chosen more often, as is natural. We use Σ to filter $\mathcal{S}$ using standard 16× anisotropic filtering.

We render all ground truth target data on-line during training. We use 256 samples per target for the prefiltering integral estimate (Equation 1) for spatially varying MatSynth materials. A larger number of samples per evaluation increases convergence speed in terms of the number of iterations but becomes prohibitively expensive for on-line generation in wall-clock time. Completely noise-free targets are not necessary as per the Noise2Noise principle [LMH*18].

4.3. Texture preconditioning

We parameterize the textures $\mathcal{S}$, $\mathcal{H}$, and $\mathcal{D}$ as dense arrays. As a preconditioner, we apply a small blurring kernel (in spatial/angular directions) to the raw optimization variables before using their values in the forward pass, similar in spirit to Nicolet et al. [NJJ21]. Unlike an explicit smoothness penalty, this does not change the optimum, but still encourages the neighboring texture entries to follow similar paths when it is beneficial. We construct the kernel in the frequency domain, using a weighting with a $1/(1 + A\sqrt{f_x^2 + f_y^2})$ falloff. In our experiments we use $A = 10$ for preconditioning the spatial texture $\mathcal{S}$ and $A = 5$ for preconditioning the angular texture $\mathcal{H}, \mathcal{D}$ when applicable (including spatially homogeneous BRDFs).

4.4. Loss function

Some previous work directly fits the logarithm and/or uses the L_1 distance [SRRW21, ZRW*24]. This is not applicable in our case, as the logarithmic transform renders the stochastic estimator of the pixel filter integral biased, violating the requirements of the Noise2Noise principle that we rely on. To circumvent this issue, we use a mixture prediction-relative L_2 and pure L_2 loss in training all models:

$$L(\omega_o, \omega_i, x, \Sigma, \Theta) = \lambda \frac{\| f_r - \exp(g(\omega_o, \omega_i, x, \Sigma, \Theta)) \|^2}{\| \mathrm{SG}(\exp(g(\omega_o, \omega_i, x, \Sigma, \Theta))) + \epsilon \|^2} + (1-\lambda) \| f_r - \exp(g(\omega_o, \omega_i, x, \Sigma, \Theta)) \|^2, \tag{10}$$

where $\mathrm{SG}(\dots)$ denotes the stop gradient operation, $f_r = f_r(\omega_o, \omega_i, x, \Sigma)$ is the ground truth and g is a model evaluation. We choose $\epsilon = 10^{-6}$. We found the use of a small pure L_2 component necessary to avoid overly bright predictions that are not sufficiently penalized by the prediction-relative L_2. To this end, we use $\lambda = 0.9999$.

4.5. Implementation details

We implement all models using PyTorch, employing custom CUDA kernels for the efficient generation of the training targets through Monte Carlo integration. We use the Adam optimizer [Kin14] with parameters $\beta_1 = 0.99$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-15}$ for all models and all materials. We use an initial learning rate of 3×10^{-3}, decayed by 0.9 every 10000 iterations. Batch size is set to 2^{16} for MatSynth materials and 2^{18} for MERL/RGL materials for all models. We apply an exponentially moving average (EMA) to all parameters with smoothing factor 0.005. Training time for our model varies from around 30 minutes per MERL/RGL (∼100 k iterations) material to 4-5 hours for MatSynth SVBRDFs (∼ 500 k iterations), trained on a single NVIDIA V100 GPU.

5. Evaluation

We perform evaluation in two parts, with the first dedicated to spatially uniform materials and the latter to materials with variations over space and scale.

Evaluation metrics. In addition to visual evaluations, we provide numerical metrics in four flavors: ꟻLIP [ANA*20], SSIM [WBSS04], root mean square error (RMSE), and mean absolute error (MAE). We compute all image-based metrics in the LDR sRGB color space. We also provide HDR metrics computed from random points from the $\omega_i, \omega_o, u, v, \Sigma$ configuration space, generated using the same method as the training data.

Baseline implementation. Our aim is to study the effects of the architectural choices employed by different models in an apples-to-apples setting. To this end, we re-implement all baselines in a common codebase, and standardize various small details to agree between the models:

- we use sigmoid linear units (SiLU) as the non-linearities;
- we always apply the exponential function as the last step, making all models output the logarithm of the reflectance function. This stabilizes training, as it removes the need for the MLP to output very large values directly;
- we optimize all latent textures, both spatial and angular, directly, using the preconditioner decribed in Section 4.3. We do not employ the encoder-driven multi-stage training process of Zeltner et al. [ZRW*24]. We find our preconditioner results in reliable, smooth convergence across model variations;
- we set the capacity of the decoding neural networks MLP_θ by choosing the number of hidden layers L and hidden channels c

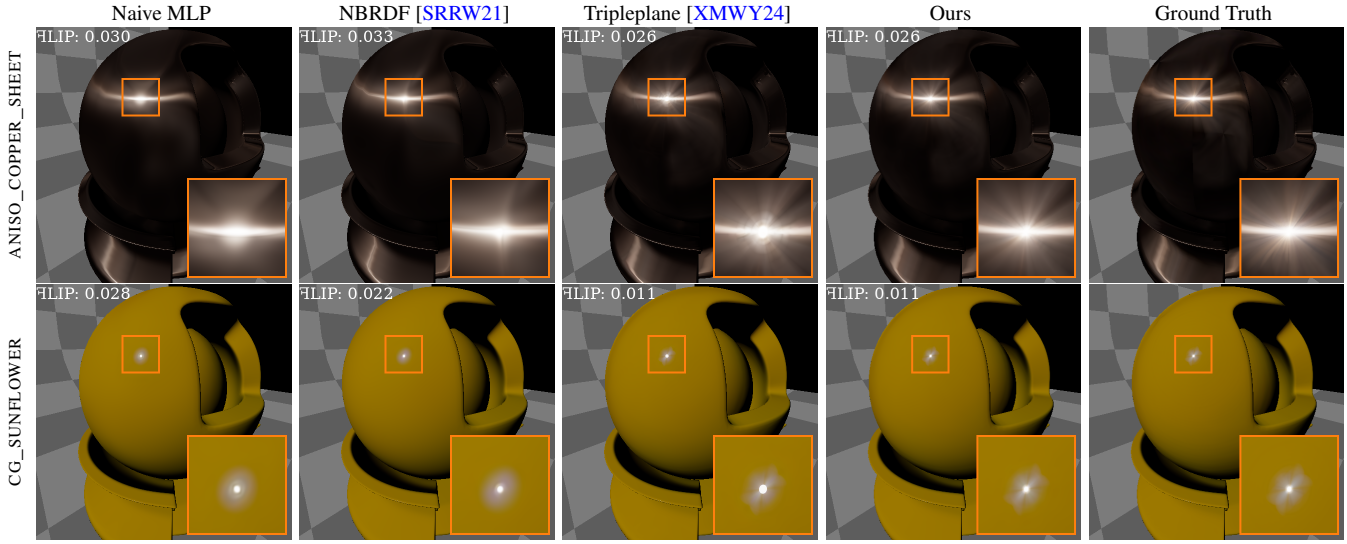


Figure 3: *Approximations of MERL and RGL materials using different methods. Methods that directly apply an neural network to light and view directions (Naive MLP and NBRDF) struggle to represent the potentially complex shape of the specular highlight. Tabulation of angular features (tripleplane and ours) greatly improves visual results and metrics. Our method captures very peaky specular highlight better (*CG_SUNFLOWER*) than the tripleplane method due to adaptivity in the angular resolution.*

of each model to roughly equalize the number of floating point operations (FLOPs) needed for a single evaluation of the reflectance model. We also adjust the angular texture dimensions D_h, D_d based on the model type and whether the gating-type conditioning (Equation 6) is active or not, aiming for equal FLOPs and a roughly similar order of magnitude in the number of parameters in the angular textures. We estimate the FLOPs from PyTorch models using a publicly available code [KAL*24];

- each model is trained using the same data sets, as described in Section 4.

5.1. Spatially homogeneous BRDFs

Baselines. We compare our method to three neural baselines:

1. a naive MLP that simply takes the incident and outgoing directions ω_i, ω_o and returns the logarithm of the reflectance value;
2. a more refined version of the first that parameterizes the directions input to the MLP with the Rusinkiewicz coordinates ω_h, ω_d, as originally suggested by Sztrajman et al. [SRRW21] as the Neural-BRDF (NBRDF);
3. the tripleplane model of Xu et al. [XMWY24], which makes use of tabulated neural features in both ω_h and ω_d. As spatial variation is not present, its main difference to our model is the lack of the learned radial warp to help better utilize the resolution in the angular latent textures.

All three baselines and our model perform 800-900 FLOPs per single inference call. Memory and compute requirements are compiled in Table 3. For additional details on parameter distribution, see Table A.8 in the supplementary material.

Figure 3 compares the models and the ground truth in two challenging homogeneous materials with complex highlight shapes. Additional results can be found in supplementary material (Figure A.17). Table 4 presents corresponding error metrics averaged over a set of 30 different homogeneous materials. As can be seen by comparing the Naive MLP to NBRDF, the move to Rusinkiewicz coordinates improves the highlight shape, and results in consistent improvements in all evalutation metrics. Tabulation of angular features in the tripleplane model further improves visual quality and all metrics. However, due to limited resolution in the angular feature maps, the tripleplane methods sometimes struggles to represent complex highlight shapes. Our model yields values of the FLIP metric that are essentially identical to those of the tripleplane, but is better at representing complex highlights due to the learned warps (Equation 7). This also manifests as the clearly best RMSE value when measured in HDR (Table 4). Figure A.10 in the supplementary material visualizes how our learned warps lead to increased

Table 3: *Compute and memory requirements of each model targeting spatially homogeneous materials assuming single precision model parameters. For reference, raw MERL data is around 35 MB and (non-spectral) RGL either 4.5 MB or 660 kB per material.*

Model configuration	FLOPs / 1	MLPs / kB	$\mathcal{H}+\mathcal{D}$ / kB	Total / kB
Naive MLP	831	2.70	0	2.70
NBRDF [SRRW21]	791	2.25	0	2.25
Tripleplane [XMWY24]	888	1.75	49.1	50.9
Ours	842	1.44	49.1	50.6

Table 4: *Image-based metrics computed from the rendering configuration of Figure 3 over 30 MERL/RGL materials. HDR metrics are computed over 2^{20} random configurations in the ω_i, ω_o space.*

Model configuration	SSIM ↑	ꟻLIP ↓	RMSE ↓	MAE ↓	RMSE-HDR ↓	MAE-HDR ↓
Naive MLP	0.9928	0.0184	0.0055	0.0024	1.0743	0.0024
NBRDF [SRRW21]	0.9939	0.0158	0.0044	0.0020	0.6534	0.0020
Tripleplane [XMWY24]	**0.9974**	**0.0104**	0.0029	**0.0012**	0.8144	**0.0012**
Ours	0.9972	0.0107	**0.0028**	**0.0012**	**0.4801**	**0.0012**

resolution in the angular textures $\mathcal{H}, \mathcal{D}$ near the highlight, thereby increasing fidelity. Overall, our method achieves close fits, and is, in many cases, visually nearly indistinguishable from the ground truth.

5.2. Spatially Varying BRDFs

Figure 1 illustrates the subtle complexities of antialiasing manifesting as complex reflectance already at relatively small viewing distances. Classical parametric models clearly fail at these setups.

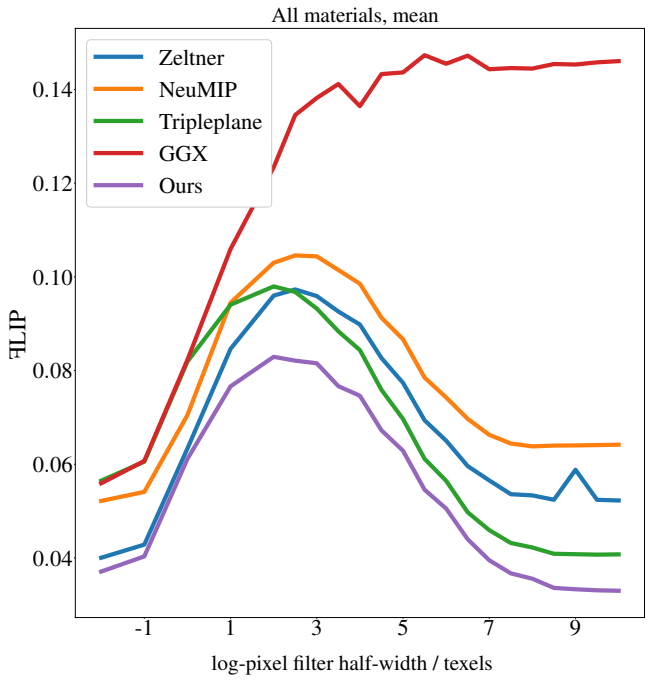


Figure 4: ꟻLIP *metric computed over a set of planar renderings with static view direction and varying directional light, using the planar scene of Figure A.11 (supplementary material).*

Baselines. With SVBRDFs, we employ the following baselines:

1. the NeuMIP model of Kuznetsov et al. [KMX*21] that learns a spatial mip-map and uses an MLP as the angular decoder. We implement their "offset module", but as our data does not contain complex BTF-like mesostructure where parallax plays a significant role, the upside may be limited;
2. the model of Zeltner et al. [ZRW*24] that also uses an MLP as the angular decoder, but in contrast to NeuMIP, further employs learned shading frames;
3. the tripleplane model of Xu et al. [XMWY24] that makes use of learned textures over ω_h, ω_d, adapted to include a mip-mapped spatial texture like the models of Zeltner and Kuznetsov. (The tripleplane model is originally designed for BTF representation at a single scale.)
4. we directly optimize a mip-map of GGX [WMLT07] parameters (specular and diffuse colors, normals, anisotropic roughness matrix; 12 channels in total) for each material to establish a baseline for what analytical microfacet models are capable of. We optimize the parameters in a training step not much unlike in recent prefiltering work [GFL*22, KAL23].

All neural models perform approx. 1600 FLOPs per single inference call, not counting work done by the texture units to fetch the spatial latent vectors, but counting angular texture reading when applicable. Memory and compute requirements are given in Table 5 and additional details on network size in Table A.9 in the supplementary material.

Metrics. The per-scaleꟻ LIP graphs averaged over our set of 25 SVBRDFs (Figure 4) confirm this numerically, clearly highlighting the need for a more expressive angular model starting already at close to medium viewing distances. All neural baselines perform much better than GGX, as expected. However, our model yields consistently the besꟻ LIP results. See Table 6 for averages of the other error metrics.

The curves in Figure 4 exhibit a characteristic shape arising from the complexity of the filtered material. At small filter scales, the ground truth BRDF reduces into the simple underlying parametric model, yielding relatively good fits for all models. Increasing the filter scale introduces complex angular behavior, making fitting more challenging. However, once the filter is large enough to include essentially the whole underlying SVBRDF texture, spatial variation in the apparent BRDF decreases, again simplifying the fitting and resulting in lower errors. Methods that employ the shading frame module (Zeltner et al., ours) are clearly better than other methods at close view distances, highlighting the benefits of such a "graphics prior". We provide additional evidence for this through ablations in the supplementary material (Section A.1). For medium to large viewing distances, with filter scale ranging from a few texels to the entire texture, models that tabulate the angular domain, tripleplane and ours, provide much sharper results and retain the angular complexity of the ground truth more faithfully than the MLP-based models (Zeltner et al. and NeuMIP).

Visual evaluation. Figure 5 compares the models visually under environmental illumination. All neural baselines yield visibly better fits than the GGX baseline, and are able to capture simple multimodalities (e.g. JS_SCIFI_PANELS_013) appearing at

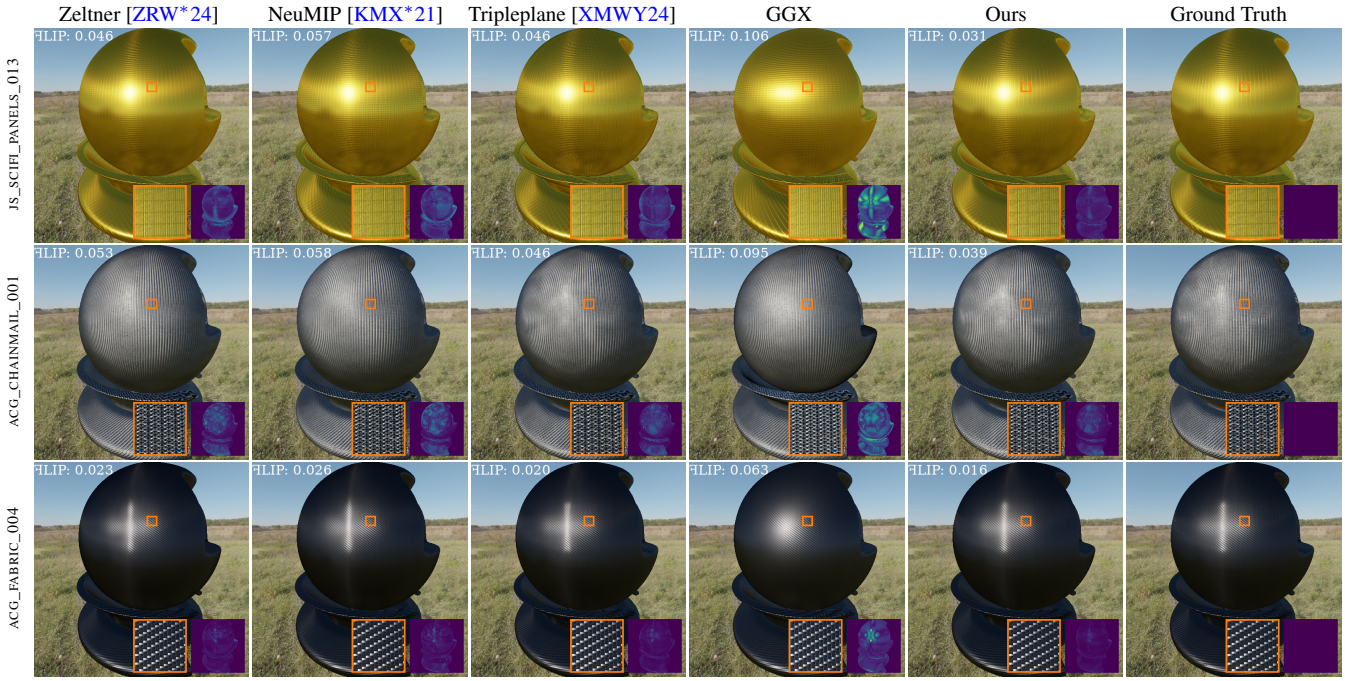


Figure 5: *Models trained on MatSynth materials under environmental lighting (all methods $2^{14}spp$). The trained models require an integral over the hemisphere whereas the ground truth also requires a nested antialiasing integral. We use MIS between the light and the ground truth analytic sampler for all images.* FLIP *is computed against the ground truth of each row and also visualized as the errormap. The GGX baseline fails at representing the complex reflectance resulting from filtering (all rows), even though at close range (insets) the parametric model is sufficient. Neural material models employing only MLPs (Zeltner, NeuMIP) are able to capture the complexity of larger filter sizes with some materials (*JS_SCIFI_PANELS_013, ACG_FABRIC_004*) but fail with others (*ACG_CHAINMAIL_001*). Models employing angular tabulation (tripleplane, ours) capture very complex filtered appearance (*ACG_CHAINMAIL_001*) much more faithfully. Our model consistently yields the best error metrics.*

medium to far viewing distances. Differences emerge when the apparent BRDF has very complex angular detail, as for instance in ACG_CHAINMAIL_001. Our model has the most representational power due to to the gating mechanism and shading frames, as confirmed in the supplementary material (Section A.1), leading to the best performance across all scales. Table 6 collects all error and similarity metrics, aggregated over materials, scales and lighting conditions. Additional materials are shown in the supplementary material (Figure A.18).

Figure A.9 in the supplementary material provides a complementary view on the complexities of the filtered apparent BRDFs, visualizing slices of the SVBRDFs in the angular domain. Figure A.12 in the supplementary material provides further visualizations with comparisons between methods.

Lastly, we present a video (please refer to the supplementary material) illustrating our model trained on various MatSynth materials and evaluated over many scales.

5.3. Performance

As a proof of concept, we have implemented our model in an OpenGL pixel shader, packing the textures $\mathcal{S}$, $\mathcal{H}$, and $\mathcal{D}$ into FP32 texture maps. As a performance baseline, we run a GGX shader using three FP16 texture maps for its 12 input channels.

Table 5: *Compute and memory requirements of each SVBRDF model. Memory is measured in kB and assumes all values are single precision floating point. The GGX model has more channels in the spatial latent than the other models (12 vs. 8). NeuMIP requires additional spatial texture channels for the neural offset module. In all cases the spatial texture dominates memory consumption even if it was quantized.*

Model configuration	FLOPs / 1	MLPs / kB	$\mathcal{H}+\mathcal{D}$ / kB	$\mathcal{S}$ (inc. MIPs) / kB	Total / kB
Zeltner [ZRW*24]	1645	4.86	0	44700	44700
NeuMIP [KMX*21]	1633	5.06	0	74100	74100
Tripleplane [XMWY24]	1638	3.98	65.5	44700	44800
GGX	~ 100	0	0	67100	67100
Ours	1625	1.79	328	44700	45100

On an NVIDIA RTX 3090 GPU, our implementation renders a 1080p image at approx. 1100 frames per second when viewing a full screen quadrilateral, with slight variation with varying mip map level. The reference GGX renderer is $2.5-10\times$ faster, depending on the mip map scale. Its $10\times$ advantage is only exhibited when viewing far-away mip map levels, which naturally access fewer tex-

Table 6: *Image-based metrics computed from 25 MatSynth materials, averaged over materials, 21 scales, 4 light directions and 3 camera positions. We use the planar scene configuration of Figure A.11 (supplementary material). HDR metrics are computed over 2^{20} random configurations in the space of $\omega_i, \omega_o, u, v, \sigma$.*

Model configuration	SSIM ↑	ꟻLIP ↓	RMSE ↓	MAE ↓	RMSE-HDR ↓	MAE-HDR ↓
Zeltner [ZRW*24]	0.9029	0.0684	0.0312	0.0194	0.7969	0.0186
NeuMIP [KMX*21]	0.8990	0.0778	0.0338	0.0212	0.9390	0.0238
Tripleplane [XMWY24]	0.9020	0.0648	0.0321	0.0208	0.8688	0.0211
GGX	0.8632	0.1296	0.0521	0.0348	6.2211	0.0527
Ours	**0.9134**	**0.0541**	**0.0272**	**0.0168**	**0.6200**	**0.0164**

Table 7: *Importance samplers in prior work. Only the work of Zeltner et al. has out-of-the-box support for SVBRDFs and multi-scale models.*

Model	Type	SVBRDFs	multi-scale
Zeltner [ZRW*24]	analytic	yes	yes
NBRDF [SRRW21]	analytic	no	no
NeuSample [XWH*23]	flow/diffusion	yes	no
Diffusion [FBL*24]	flow/diffusion	no	no
Reparametrization [WBX*25]	direct warp	yes	no
Importance baking [BWZ*25]	direct warp	no	no

els over the entire image, leading to better texture cache utilization and shifting the memory/compute balance.

While both implementations are straightforward and unoptimized, with likely gains available from careful choice of bit depths, quantization, etc., the results indicate surprisingly high performance in both absolute terms and relative to the GGX baseline.

6. Analysis

6.1. Anisotropic filtering

We train all SVBRDF models with anisotropically filtered targets, and employ corresponding anisotropic texture fetches during the forward passes. Figure 6 visualizes the benefits. A model trained with only isotropic targets and evaluated without anisotropic filtering (left) yields overly blurry results, as expected. Training the model with only isotropic targets but using anisotropic filtering during inference (2nd left) clears most of the blur, but does not yield the correct result, with many specular highlights missing. Training with anisotropic targets in the dataset as well as applying anisotropic filtering during inference (2nd right) gives results much closer to the ground truth (right).

6.2. Curvature and visibility

Our model is built on the assumptions that the shaded surface is locally planar, and that the pixel footprint completely covers the surface. Figure 7 illustrates a scene with many visibility discontinuities, and where the curvature of geometry under the pixel footprints starts to become significant. We render the ground truth image by Monte Carlo integration in the image domain, which accounts for curvature and antialiases visibility. The quasi ground truth image is rendered with a single sample through the center of each pixel, performing the antialiasing in texture space according to Equation 1 using Monte Carlo integration. It represents the best possible result obtainable under our assumption of local planarity at a single sample per pixel. Our result using a single sample per pixel (2nd right) closely matches the quasi ground truth. For completeness, we also show multi-sample antialiasing (MSAA, right) where shading is computed once per pixel per primitive using our model, but visibility is supersampled by Monte Carlo integration. This image corresponds closely to a practical use scenario in real-time rendering.

Despite these results, it is likely that very sharp curvature will cause visible differences to the actual ground truth. Therefore, it would be interesting to extend the model to curvature-aware reflectance prediction [KWM*22].

6.3. Limitations

We observe some discrepancy in color in materials that have rare splotches of color differing from the dominant color of the material. ACG_CHAINMAIL_001 contains rusty parts sparsely scattered around the material that are not captured by our model (see Figure 8, left). This could likely be addressed by rebalancing the training samples or adjusting the loss further.

Materials whose prefiltered apparent BRDFs exhibit extreme complexity (Figure 8, right) are the worst case for any approximate model, including ours. The material shown represents a crumpled foil, a shiny metal surface with significant normal variations across the spatial coordinates.

Since our model primarily relies on (bilinear) interpolation also in the angular domain, it is prone to the customary banding artifacts related to the discontinuity of derivatives of the bilinear reconstruction kernel. Such artifacts can be seen, for example, in CGBC_M.WEAVE_2 (ours, red panel) in Figure A.9 in the supplementary material. While computationally and memory bandwidth-wise more intensive, this could in principle be remedied by higher-order interpolation schemes. The same figure also shows artifacts arising from the choice of polar coordinates.

Even though we do not provide an importance sampler in this work it does not mean that samplers introduced in prior work could not be employed. In fact, as described in Section 2 the importance sampler is almost always independent and separate from the reflectance model. Table 7 collects properties of importance samplers in prior work. Our reflectance model requires a sampler that supports spatial variation and variation over scale which is only satisfied by the model of Zeltner at al. However, their analytic GGX-based sampler may yield very poor fits to many materials in our training set (as demonstrated by our results with GGX as one of the baselines). Hence it may be required to extend one of the more expressive SVBRDF-tested importance samplers (e.g. NeuSample [XWH*23]) to also support multiple scales.

7. Conclusions and Future Work

We have presented a neural reflectance model that employs textures in both angular and spatial domains, and demonstrated its effectiveness in both spatially uniform and spatially varying materials. The model can be run in true real time, with the execution time of our proof-of-concept implementation within an order of magnitude of a standard analytical GGX SVBRDF model. Our model supports multiple levels of detail, gracefully handling complex angular effects induced by pixel filtering.

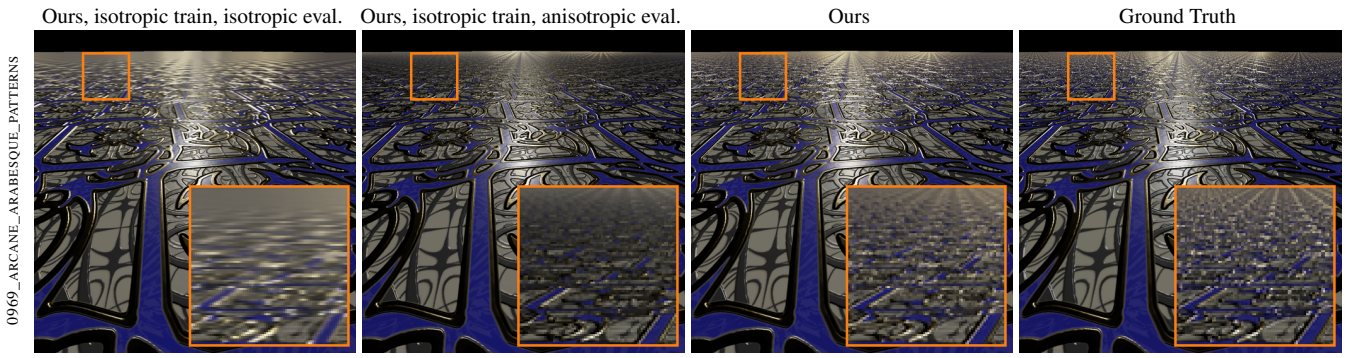


Figure 6: *Anisotropic spatial texture filtering. Only showing isotropically filtered targets and evaluating the model with isotropic filtering unsurprisingly leads into excessive blurring (first column) at low viewing angles. Training the model with isotropic targets, but filtering the spatial texture with anisotropic filtering during evaluation reduces blurring, but misses specular highlights (second column). Showing anisotropically filtered targets to the model and evaluating the model using anisotropic filtering (third column) yields results closer to ground truth. We render the ground truth stochastically at* 2^{16} *spp using the highest resolution texture maps.*

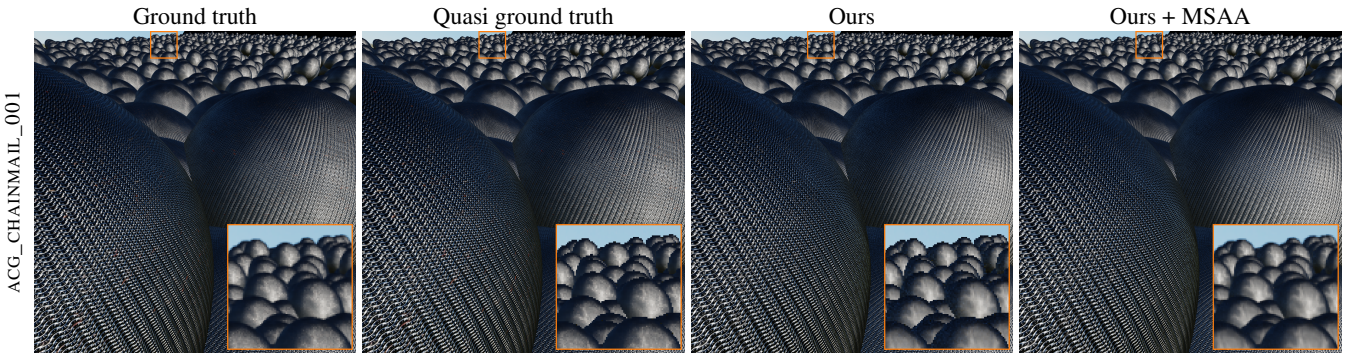


Figure 7: *Significant curvature on pixel scale. Ground truth rendered using supersampling (left) and with planar-approximated filtering for texture only (second from the left "quasi ground truth"). Curvature causes a visible but not very dramatic difference between the two methods. Our model (second from the right) is trained as the quasi ground truth as the target and the image quality is likely limited more by errors in the fitting process (ours–quasi ground truth) than by the error caused by ignoring curvature. On the right we show our model integrated with MSAA, where visibility is supersampled but the expensive texture filtering is offloaded to our multi-scale neural material model and evaluated with a single sample per pixel.*

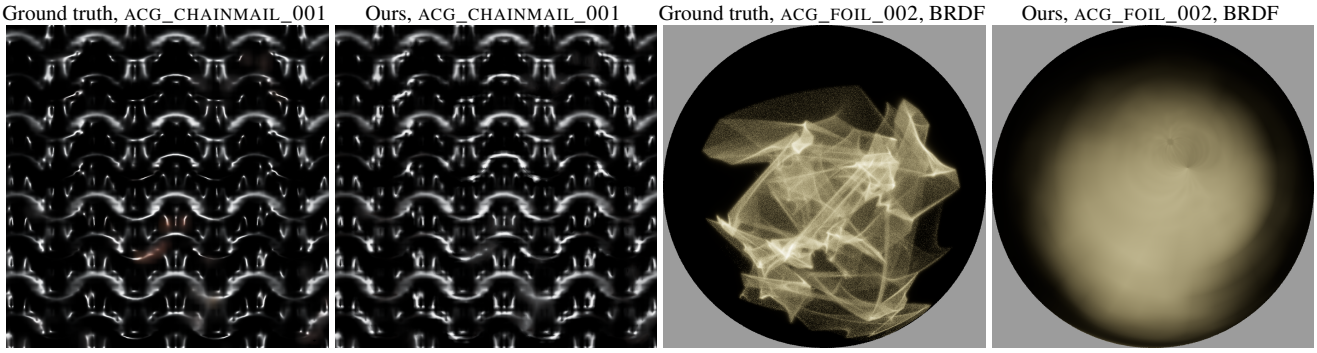


Figure 8: *Limitations. Left: missing colors. Our model sometimes misses rare patches of color differing from the dominant color of the material.* ACG_CHAINMAIL_001 *shows rusty (diffuse) parts which are shown as light gray with our model (i.e. overly bright in some channels). Right: shiny materials that exhibit complex normal variation produce filtered BRDFs of very fine detail (here a metallic foil,* ACG_FOIL_002*). None of the models, including ours, is able to capture the complexity.*

Future work. In this work, we have focused only on the evaluation of the BRDF, without considering its importance sampling at all. Extending the model to also include an appropriate importance sampler is an immediate next step for future work. Theoretical work also remains as we do not have guarantees for energy conservation, an important property to guarantee physical plausibility. The projection from the hemisphere to the disk and interpolation of features on a plane may cause distortions and hurt model performance which requires more thorough future study. Finally, the shader implementation of our model is unoptimized, and likely does not require full precision floating point numbers to look plausible. We leave such performance engineering and optimization for future work.

8. Acknowledgments

MatSynth materials labeled with the *js* prefix are by Julio Sillet and licensed under CC-BY. The Mitsuba knob used in many of the renderings is reproduced with permission. Environmental lights used in some of the figures are from polyhaven.com and licensed under CC0. We would like to thank the anonymous reviewers for their feedback as well as Erik Härkönen for GPU implementations of the MERL and RGL material models. Computational resources used in this work were provided by the Aalto Science-IT project. This work was partially supported by the European Research Council (ERC Consolidator Grant 866435). Open access publishing facilitated by Aalto-yliopisto, as part of the Wiley - FinELib agreement.

References

[ANA*20] ANDERSSON P., NILSSON J., AKENINE-MÖLLER T., OSKARSSON M., ÅSTRÖM K., FAIRCHILD M. D.: ꟻLIP: A Difference Evaluator for Alternating Images. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 3*, 2 (2020), 15:1–15:23. 6

[APS00] ASHIKMIN M., PREMOŽE S., SHIRLEY P.: A microfacet-based brdf generator. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., p. 65–74. URL: https://doi.org/10.1145/344779.344814, doi:10.1145/344779.344814. 2

[AS00] ASHIKHMIN M., SHIRLEY P.: An anisotropic phong brdf model. *Journal of graphics tools 5*, 2 (2000), 25–32. 2

[BCB15] BAHDANAU D., CHO K., BENGIO Y.: Neural machine translation by jointly learning to align and translate. In *Proc. International Conference on Learning Representations (ICLR) 2015* (2015). 4

[Bli77] BLINN J. F.: Models of light reflection for computer synthesized pictures. In *Proceedings of the 4th annual conference on Computer graphics and interactive techniques* (1977), pp. 192–198. 2

[Bli78] BLINN J. F.: Simulation of wrinkled surfaces. *SIGGRAPH Comput. Graph. 12*, 3 (Aug. 1978), 286–292. URL: https://doi.org/10.1145/965139.507101, doi:10.1145/965139.507101. 3

[BM93] BECKER B. G., MAX N. L.: Smooth transitions between bump rendering algorithms. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1993), SIGGRAPH '93, Association for Computing Machinery, p. 183–190. URL: https://doi.org/10.1145/166117.166141, doi:10.1145/166117.166141. 3

[BN11] BRUNETON E., NEYRET F.: A survey of nonlinear prefiltering methods for efficient and accurate surface shading. *IEEE Transactions on Visualization and Computer Graphics 18*, 2 (2011), 242–260. 3

[BS12] BURLEY B., STUDIOS W. D. A.: Physically-based shading at disney. In *ACM Siggraph* (2012), vol. 2012, vol. 2012, pp. 1–7. 2

[BSH12] BAGHER M. M., SOLER C., HOLZSCHUCH N.: Accurate fitting of measured reflectances using a shifted gamma micro-facet distribution. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 1509–1518. 2

[BWZ*25] BAI Y., WU S., ZENG Z., YAN L., WANG B.: Brdf importance baking: A lightweight neural solution to importance sampling general parametric brdfs. *Computer Graphics Forum n/a*, n/a (2025), e70286. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.70286, doi:https://doi.org/10.1111/cgf.70286. 4, 10

[CFW*21] CAO Y., FANG Z., WU Y., ZHOU D.-X., GU Q.: Towards understanding the spectral bias of deep learning. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)* (2021). 3

[CMS87] CABRAL B., MAX N., SPRINGMEYER R.: Bidirectional reflection functions from surface bump maps. *SIGGRAPH Comput. Graph. 21*, 4 (Aug. 1987), 273–281. URL: https://doi.org/10.1145/37402.37434, doi:10.1145/37402.37434. 3

[CT82] COOK R. L., TORRANCE K. E.: A reflectance model for computer graphics. *ACM Transactions on Graphics (ToG) 1*, 1 (1982), 7–24. 2

[CXG*22] CHEN A., XU Z., GEIGER A., YU J., SU H.: Tensorf: Tensorial radiance fields. In *Computer Vision – ECCV 2022* (Cham, 2022), Avidan S., Brostow G., Cissé M., Farinella G. M., Hassner T., (Eds.), Springer Nature Switzerland, pp. 333–350. 4

[DHI*13] DUPUY J., HEITZ E., IEHL J.-C., POULIN P., NEYRET F., OSTROMOUKHOV V.: Linear efficient antialiased displacement and reflectance mapping. *ACM Trans. Graph. 32*, 6 (Nov. 2013). URL: https://doi.org/10.1145/2508363.2508422, doi:10.1145/2508363.2508422. 3

[DJ18] DUPUY J., JAKOB W.: An adaptive parameterization for efficient material acquisition and rendering. *ACM Transactions on graphics (TOG) 37*, 6 (2018), 1–14. 2, 5

[DMD*23] DATTA S., MARSHALL C., DONG Z., LI Z., NOWROUZEZAHRAI D.: Efficient graphics representation with differentiable indirection. In *SIGGRAPH Asia 2023 Conference Papers* (New York, NY, USA, 2023), SA '23, Association for Computing Machinery. URL: https://doi.org/10.1145/3610548.3618203, doi:10.1145/3610548.3618203. 4

[DZJ*24] DOU Y., ZHENG Z., JIN Q., NI B., CHEN Y., KE J.: Real-time neural brdf with spherically distributed primitives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 4337–4346. 3

[FBL*24] FU Z., BELHE Y., LU H., WU L., XU B., LI T.-M.: Bsdf importance sampling using a diffusion model. In *SIGGRAPH Asia 2024 Conference Papers* (New York, NY, USA, 2024), SA '24, Association for Computing Machinery. URL: https://doi.org/10.1145/3680528.3687684, doi:10.1145/3680528.3687684. 4, 10

[Fou92] FOURNIER A.: *Filtering Normal Maps and Creating Multiple Surfaces*. Tech. rep., CAN, 1992. 3

[FWH*23] FAN J., WANG B., HASAN M., YANG J., YAN L.-Q.: Neural biplane representation for btf rendering and acquisition. In *ACM SIGGRAPH 2023 Conference Proceedings* (2023), pp. 1–11. 3

[GFL*22] GAUTHIER A., FAURY R., LEVALLOIS J., THONAT T., THIERY J.-M., BOUBEKEUR T.: Mipnet: Neural normal-to-anisotropic-roughness mip mapping. *ACM Transactions on Graphics 41*, 6 (2022), 1–12. 3, 8

[Gou71] GOURAUD H.: Continuous shading of curved surfaces. *IEEE Trans. Comput. 20*, 6 (June 1971), 623–629. URL: https://doi.org/10.1109/T-C.1971.223313, doi:10.1109/T-C.1971.223313. 2

[Hec89] HECKBERT P. S.: *Fundamentals of Texture Mapping and Image Warping*. Tech. rep., USA, 1989. 2

[HSRG07] HAN C., SUN B., RAMAMOORTHI R., GRINSPUN E.: Frequency domain normal map filtering. In *ACM SIGGRAPH 2007 Papers* (New York, NY, USA, 2007), SIGGRAPH '07, Association for Computing Machinery, p. 28–es. URL: https://doi.org/10.1145/1275808.1276412, doi:10.1145/1275808.1276412. 3

[HX23] HAN K., XIANG W.: Multiscale tensor decomposition and rendering equation encoding for view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2023), pp. 4232–4241. 4

[KAL23] KEMPPINEN P., AITTALA M., LEHTINEN J.: Data-driven pixel filter aware mip maps for svbrdfs. In *Eurographics Symposium on Rendering* (Netherlands, 2023), Ritschel T., Weidlich A., (Eds.), Eurographics Association, pp. 51–61. Eurographics Symposium on Rendering, EGSR ; Conference date: 28-06-2023 Through 30-06-2023. URL: https://conferences.eg.org/egsr2023/, doi:10.2312/sr.20231124. 3, 8

[KAL*24] KARRAS T., AITTALA M., LEHTINEN J., HELLSTEN J., AILA T., LAINE S.: Analyzing and improving the training dynamics of diffusion models, 2024. URL: https://github.com/NVlabs/edm2/blob/main/count_flops.py. 7

[Kin14] KINGMA D. P.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 6

[KLA19] KARRAS T., LAINE S., AILA T.: A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 4401–4410. 4

[KMX*21] KUZNETSOV A., MULLIA K., XU Z., HAŠAN M., RAMAMOORTHI R.: NeuMIP: multi-resolution neural materials. *ACM Trans. Graph. 40*, 4 (July 2021). URL: https://doi.org/10.1145/3450626.3459795, doi:10.1145/3450626.3459795. 2, 3, 8, 9, 10

[KWM*22] KUZNETSOV A., WANG X., MULLIA K., LUAN F., XU Z., HASAN M., RAMAMOORTHI R.: Rendering neural materials on curved surfaces. In *ACM SIGGRAPH 2022 Conference Proceedings* (New York, NY, USA, 2022), SIGGRAPH '22, Association for Computing Machinery. URL: https://doi.org/10.1145/3528233.3530721, doi:10.1145/3528233.3530721. 10

[LFTG97] LAFORTUNE E. P. F., FOO S.-C., TORRANCE K. E., GREENBERG D. P.: Non-linear approximation of reflectance functions. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., p. 117–126. URL: https://doi.org/10.1145/258734.258801, doi:10.1145/258734.258801. 2

[LGZL*20] LIU L., GU J., ZAW LIN K., CHUA T.-S., THEOBALT C.: Neural sparse voxel fields. *Advances in Neural Information Processing Systems 33* (2020), 15651–15663. 4

[LKYU12] LÖW J., KRONANDER J., YNNERMAN A., UNGER J.: Brdf models for accurate and efficient rendering of glossy surfaces. *ACM Trans. Graph. 31*, 1 (Feb. 2012). URL: https://doi.org/10.1145/2077341.2077350, doi:10.1145/2077341.2077350. 2

[LMH*18] LEHTINEN J., MUNKBERG J., HASSELGREN J., LAINE S., KARRAS T., AITTALA M., AILA T.: Noise2noise: Learning image restoration without clean data. In *International Conference on Machine Learning* (2018), PMLR, pp. 2965–2974. 6

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG) 41*, 4 (2022), 1–15. 4

[MPBM03] MATUSIK W., PFISTER H., BRAND M., MCMILLAN L.: A data-driven reflectance model. *ACM Transactions on Graphics (TOG) 22*, 3 (2003), 759–769. 2, 5

[NDM05] NGAN A., DURAND F., MATUSIK W.: Experimental analysis of brdf models. *Rendering Techniques 2005*, 16th (2005), 2. 2

[NJJ21] NICOLET B., JACOBSON A., JAKOB W.: Large steps in inverse rendering of geometry. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia) 40*, 6 (Dec. 2021). doi:10.1145/3478513.3480501. 6

[OB10] OLANO M., BAKER D.: Lean mapping. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2010), I3D '10, Association for Computing Machinery, p. 181–188. URL: https://doi.org/10.1145/1730804.1730834, doi:10.1145/1730804.1730834. 3

[ON94] OREN M., NAYAR S. K.: Generalization of lambert's reflectance model. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1994), SIGGRAPH '94, Association for Computing Machinery, p. 239–246. URL: https://doi.org/10.1145/192161.192213, doi:10.1145/192161.192213. 2

[Pho75] PHONG B. T.: Illumination for computer generated pictures. *Commun. ACM 18*, 6 (June 1975), 311–317. URL: https://doi.org/10.1145/360825.360839, doi:10.1145/360825.360839. 2

[RGJW20] RAINER G., GHOSH A., JAKOB W., WEYRICH T.: Unified neural encoding of btfs. In *Computer Graphics Forum* (2020), vol. 39, Wiley Online Library, pp. 167–178. 2

[RJGW19] RAINER G., JAKOB W., GHOSH A., WEYRICH T.: Neural btf compression and interpolation. In *Computer Graphics Forum* (2019), vol. 38, Wiley Online Library, pp. 235–244. 2

[RPKLMG23] RODRIGUEZ-PARDO C., KAZATZIS K., LOPEZ-MORENO J., GARCES E.: Neubtf: Neural fields for btf encoding and transfer. *Computers & Graphics 114* (2023), 239–246. URL: https://www.sciencedirect.com/science/article/pii/S0097849323001140, doi:https://doi.org/10.1016/j.cag.2023.06.018. 2

[Rus98] RUSINKIEWICZ S. M.: A new change of variables for efficient brdf representation. In *Rendering Techniques' 98: Proceedings of the Eurographics Workshop in Vienna, Austria, June 29—July 1, 1998 9* (1998), Springer, pp. 11–22. 2, 4

[SRRW21] SZTRAJMAN A., RAINER G., RITSCHEL T., WEYRICH T.: Neural brdf representation and importance sampling. In *Computer Graphics Forum* (2021), vol. 40, Wiley Online Library, pp. 332–346. 2, 3, 4, 6, 7, 8, 10

[Tok05] TOKSVIG M.: Mipmapping normal maps. *Journal of graphics tools 10*, 3 (2005), 65–71. 3

[VD24] VECCHIO G., DESCHAINTRE V.: Matsynth: A modern pbr materials dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024). 5

[War92] WARD G. J.: Measuring and modeling anisotropic reflection. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques* (1992), pp. 265–272. 2

[WBSS04] WANG Z., BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing 13*, 4 (2004), 600–612. 6

[WBX*24] WU L., BI S., XU Z., LUAN F., ZHANG K., GEORGIEV I., SUNKAVALLI K., RAMAMOORTHI R.: Neural directional encoding for efficient and accurate view-dependent appearance modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 21157–21166. 4

[WBX*25] WU L., BI S., XU Z., TAN H., ZHANG K., LUAN F., LU H., RAMAMOORTHI R.: Neural brdf importance sampling by reparameterization. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers* (New York, NY, USA, 2025), SIGGRAPH Conference Papers '25, Association for Computing Machinery. URL: https://doi.org/10.1145/3721238.3730679, doi:10.1145/3721238.3730679. 4, 10

[WDR11] WU H., DORSEY J., RUSHMEIER H.: Physically-based interactive bi-scale material design. *ACM Trans. Graph. 30*, 6 (Dec. 2011), 1–10. URL: https://doi.org/10.1145/2070781.2024179, doi:10.1145/2070781.2024179. 3

[WMLT07] WALTER B., MARSCHNER S. R., LI H., TORRANCE K. E.: Microfacet models for refraction through rough surfaces. *Rendering techniques 2007* (2007), 18th. 2, 8

[XCL*25] XU Z., CHEN X., LIU C., WANG B., WANG L., MONTAZERI Z., YAN L.-Q.: Towards comprehensive neural materials: Dynamic structure-preserving synthesis with accurate silhouette at instant inference speed. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers* (New York, NY, USA, 2025), SIGGRAPH Conference Papers '25, Association for Computing Machinery. URL: https://doi.org/10.1145/3721238.3730626, doi:10.1145/3721238.3730626. 3

[XMWY24] XU Z., MONTAZERI Z., WANG B., YAN L.-Q.: A dynamic by-example btf synthesis scheme. In *SIGGRAPH Asia 2024 Conference Papers* (New York, NY, USA, 2024), SA '24, Association for Computing Machinery. URL: https://doi.org/10.1145/3680528.3687578, doi:10.1145/3680528.3687578. 2, 3, 4, 7, 8, 9, 10

[XWH*23] XU B., WU L., HASAN M., LUAN F., GEORGIEV I., XU Z., RAMAMOORTHI R.: Neusample: Importance sampling for neural materials. In *ACM SIGGRAPH 2023 Conference Proceedings* (New York, NY, USA, 2023), SIGGRAPH '23, Association for Computing Machinery. URL: https://doi.org/10.1145/3588432.3591524, doi:10.1145/3588432.3591524. 4, 10

[XWZB17] XU C., WANG R., ZHAO S., BAO H.: Real-Time Linear BRDF MIP-Mapping. *Computer Graphics Forum* (2017). doi:10.1111/cgf.13221. 3

[XXP*23] XU L., XIANGLI Y., PENG S., PAN X., ZHAO N., THEOBALT C., DAI B., LIN D.: Grid-guided neural radiance fields for large urban scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2023), pp. 8296–8306. 4

[ZRW*24] ZELTNER T., ROUSSELLE F., WEIDLICH A., CLARBERG P., NOVÁK J., BITTERLI B., EVANS A., DAVIDOVIČ T., KALLWEIT S., LEFOHN A.: Real-time neural appearance models. *ACM Transactions on Graphics 43*, 3 (2024), 1–17. 2, 3, 4, 5, 6, 8, 9, 10